DATA PAPER

CN-DBpedia2: An Extraction and Verification Framework for Enriching Chinese Encyclopedia Knowledge Base

Bo Xu[1], Jiaqing Liang[2], Chenhao Xie[2], Bin Liang[2], Lihan Chen[2] & Yanghua Xiao[2†]

[1]School of Computer Science and Technology, Donghua University, Shanghai 200051, China

[2]Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University, Shanghai 200433, China

Keywords: Knowledge graph; Entity typing; Slot filling; Information extraction; Crowdsourcing

Citation: B. Xu, J. Liang, C. Xie, B. Liang, L. Chen & Y. Xiao. CN-DBpedia2: An extraction and verification framework for enriching Chinese encyclopedia knowledge base. Data Intelligence 1(2019), 244-261. doi: 10.1162/dint_a_00017
Received: January 4, 2019; Revised: May 14, 2019; Accepted: May 24, 2019

ABSTRACT

Knowledge base plays an important role in machine understanding and has been widely used in various applications, such as search engine, recommendation system and question answering. However, most knowledge bases are incomplete, which can cause many downstream applications to perform poorly because they cannot find the corresponding facts in the knowledge bases. In this paper, we propose an extraction and verification framework to enrich the knowledge bases. Specifically, based on the existing knowledge base, we first extract new facts from the description texts of entities. But not all newly-formed facts can be added directly to the knowledge base because the errors might be involved by the extraction. Then we propose a novel crowd-sourcing based verification step to verify the candidate facts. Finally, we apply this framework to the existing knowledge base *CN-DBpedia* and construct a new version of knowledge base *CN-DBpedia2*, which additionally contains the high confidence facts extracted from the description texts of entities.

1. INTRODUCTION

In recent years, there has been a great amount of efforts in trying to harvest knowledge from Web, and a variety of *knowledge graphs* (KGs) or *knowledge bases* (KBs) have been constructed, such as YAGO [1], DBpedia [2], Freebase [3] and CN-DBpedia [4]. These knowledge bases play important roles in many applications, such as search engine [5], recommendation system [6] and question answering [7].

† Corresponding author: Yanghua Xiao (Email: shawyh@fudan.edu.cn; ORCID: 0000-0001-8403-9591).

However, knowledge bases are generally incomplete. Facts of current KBs (e.g., DBpedia [8], YAGO [1], Freebase [3] and CN-DBpedia [4]) are mainly obtained from the carefully edited structured texts (e.g., infobox and category information) of Web pages in the online encyclopedia websites (e.g., Wikipedia and Baidu Baike[①]). Since knowledge is rich but editors have limited editing capabilities, many structured texts are often incomplete, making the facts in knowledge bases directly extracted from structured texts are incomplete. According to Catriple [9], only 44.2% of articles in Wikipedia have infobox information. Also in Baidu Baike, the largest Chinese online encyclopedia website, almost 32% (nearly 3 million) of entities lack the infobox and category information altogether [10].

Incomplete knowledge base will lead to poor performance of many downstream applications, since they cannot find the corresponding facts in the knowledge bases. For example, if the knowledge graphs lack the fact about *Donald Trump*'s birthday, then it cannot answer the question of "*when was Donald Trump born*".

To address this challenge, we propose an *extraction and verification* framework to enrich the knowledge bases. Based on the existing knowledge bases, we first extract new facts from the description texts of entities. But not all newly-formed facts can be added directly to the knowledge base because the errors might be involved by the extraction [11, 12, 13]. For example, in Table 1, which is the F1-score of the state-of-the-art text-based extractors on *Slot Filling* benchmark TAC data set, including the pattern-based method (PATdist [11]), traditional machine learning methods (Mintz++ [14], SVMskip [11]), graphical model (MIMLRE [15]) and neural network based method (CNNcontext) [11], the facts extracted by these extractors still have a lot of noise. This motivates us to employ a novel crowdsourcing method to verify the extracted facts. Considering the human cost, we only verify those low-confidence facts. In the end, only two types of facts extracted by the extractors can be added to the knowledge base. One is the facts with high confidence, and the other is the facts with low confidence but verified by human as correct.

Table 1. The F1 scores on slot filling benchmark TAC data set (development (dev) set: data from 2012/2013, evaluation (eval) set: data from 2014) [11].

	Mintz++		MIMLRE		PATdist		SVMskip		CNNcontext	
	dev	eval	dev	eval	dev	eval	dev	eval	dev	eval
Per:age	.84	.71	.83	.73	.69	.80	.86	.74	.83	.76
Per:alternate names	.29	.03	.29	.03	.50	.50	.35	.02	.32	.04
Per:children	.76	.43	.77	.48	.10	.07	.81	.68	.82	.61
Per:cause of death	.76	.42	.75	.36	.44	.11	.82	.32	.77	.52
Per:date of birth	1.0	.60	.99	.60	.67	.57	1.0	.67	1.0	.77
Per:date of death	.67	.45	.67	.45	.30	.32	.79	.54	.72	.48
Per:empl memb of	.38	.36	.41	.37	.24	.22	.42	.36	.41	.37
Per:location of birth	.56	.22	.56	.22	.30	.30	.59	.27	.59	.23
Per:loc of death	.65	.41	.66	.43	.13	.00	.64	.34	.63	.28
Per:loc of residence	.14	.11	.15	.18	.10	.03	.31	.33	.20	.23
Average	.53	.41	.54	.42	.35	.36	.62	.48	.60	.46

① http://baike.baidu.com/

The missing facts in the knowledge base mainly include the relationship between entities and entities and the relationship between entities and concepts. In this paper, we use description texts of entities to enrich the knowledge base, including two subtasks, *entity typing* and *slot filling*. Our contributions are as follows:

1). First, for entity typing subtask, we propose a multi-instance learning model to process textual information as well as heterogeneous information.
2). Second, for slot filling subtask, we use a transfer learning strategy to extract the values of the long-tailed predicates.
3). Third, we propose a novel implicit crowdsourcing approach to verify low-confidence new facts.
4). Finally, we apply this framework to the existing knowledge base *CN-DBpedia* and release a new version of knowledge base *CN-DBpedia*2, which additionally contains the facts extracted from the description texts of entities. In April 2019, *CN-DBpedia*2 contained about 16,024,656 entities and 228,499,155 facts.

The rest of this paper is organized as follows. Section 2 introduces the system architecture of *CN-DBpedia*2. Section 3 and Section 4 detail the methods of entity typing and slot filling. Section 5 introduces how to verify those low-confidence new facts. Section 6 presents the statistics of our new system. Finally, Section 7 concludes the paper.

2. SYSTEM ARCHITECTURE

The system architecture of *CN-DBpedia*2 is shown in Figure 1, which is an extension of *CN-DBpedia*. *CN-DBpedia*2 uses Baidu Baike, Hudong Baike, Chinese Wikipedia and other domain encyclopedia websites as data sources, and the pipeline process consists of five components:

- The extraction component is used to extract the raw facts from the articles of the encyclopedia websites, including crawling the Web pages of all the entities in the data sources, and extracting the raw facts from the structured text of the pages.
- The normalization component is used to normalize the raw facts, including the normalization of attributes/predicates and values of the facts.
- The enrichment component is used to extract new facts that cannot be obtained directly from structured text of Web pages.
- The correction component is used to correct some of the error facts in the knowledge base, including error detection and crowdsourcing error correction.
- The update component is used to keep the freshness of knowledge base, including periodic update and active update.

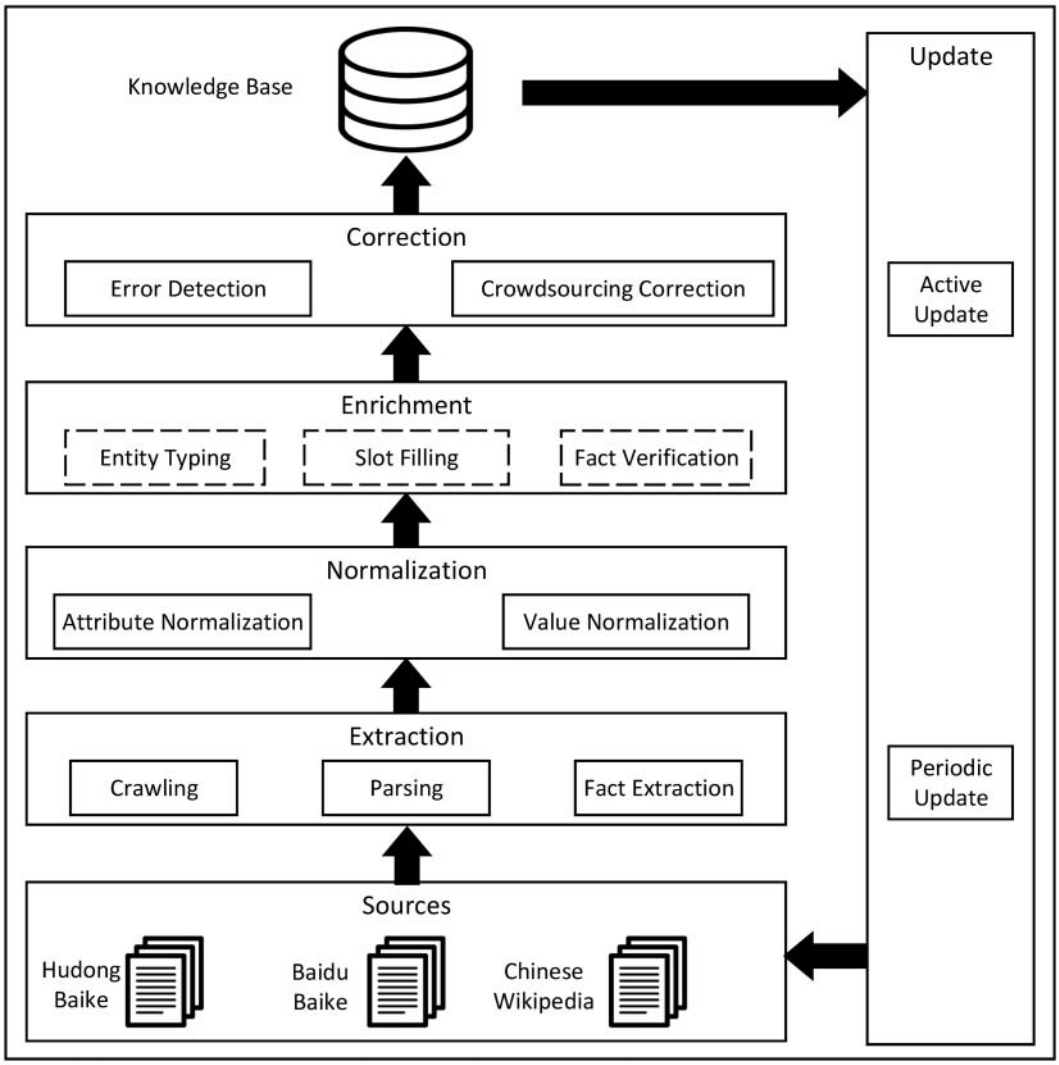


Figure 1. System architecture of *CN-DBpedia*2. Note: It is an extension of *CN-DBpedia*, and the dotted components are new features.

*CN-DBpedia*2 is different from *CN-DBpedia* in the enrichment component. We propose an extraction and verification framework to enrich the knowledge bases, which includes three new features, entity typing, slot filling and fact verification. As shown in Figure 2, we use both entity typing and slot filling methods to extract new facts from the description texts of entities, and low-confidence facts need to be verified before they are added to the knowledge base.

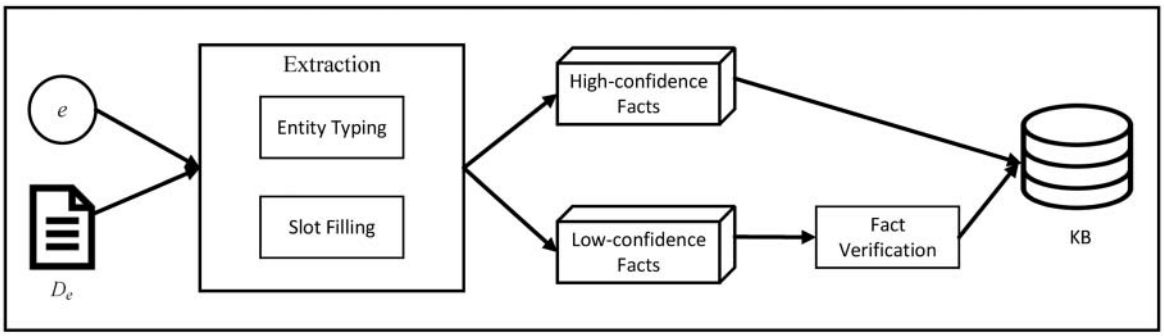


Figure 2. The detail of enrichment component in *CN-DBpedia2*.

3. ENTITY TYPING

The entity typing task is to find a set of types/concepts for each entity in knowledge bases. An entity contains both structured and unstructured features. In *CN-DBpedia*, we have used structured features to type entities [10]. In *CN-DBpedia2*, we first use unstructured features alone to type entities and then use both structured and unstructured features together to type the entities.

3.1 Entity Typing from Unstructured Features

We propose a multi-instance method, *METIC* [13] to type the entity with unstructured features alone. An entity may have multiple mentions in a corpus, we take each mention of an entity in KBs as an instance of the entity, and learn the types of these entities from multiple instances. Specifically, we first use an end-to-end neural network model to type each instance of an entity (*mention typing*), and then use an *integer linear programming* (ILP) method to aggregate the predicted type results from multiple instances (*type fusion*). The framework of our solution is shown in Figure 3.

In the offline phase, we train models for the two subtasks mention typing and type fusion separately. For mention typing, we model it as a *multi-label classification*. In our setting, we use the distant supervision method to construct the training data automatically and build a supervised learning model (more specifically we propose a neural network model). For type fusion, we model it as a constrained optimization problem, and propose an integer linear programming model in order to solve the problem. The constraints are derived from the semantic relationship between types.

In the online phase, we use the models built in the offline phase to enrich types for each entity in the KB. For each entity *e*, we first employ the existing entity linking systems [16] to discover entity mentions from its corpus. Each mention and its corresponding context: $<m_i, c_i>$ are fed into the mention typing model. The model then derives a set of types with each type being associated with a probability (i.e., $(t|m_i)$). Types as well as their probability (i.e., $P(m_i)$) derived from each mention are further fed into the integer linear programming model with constraints specified as exclusive or compatibility among types. The model finally selects a subset from all candidate types as the final output types.

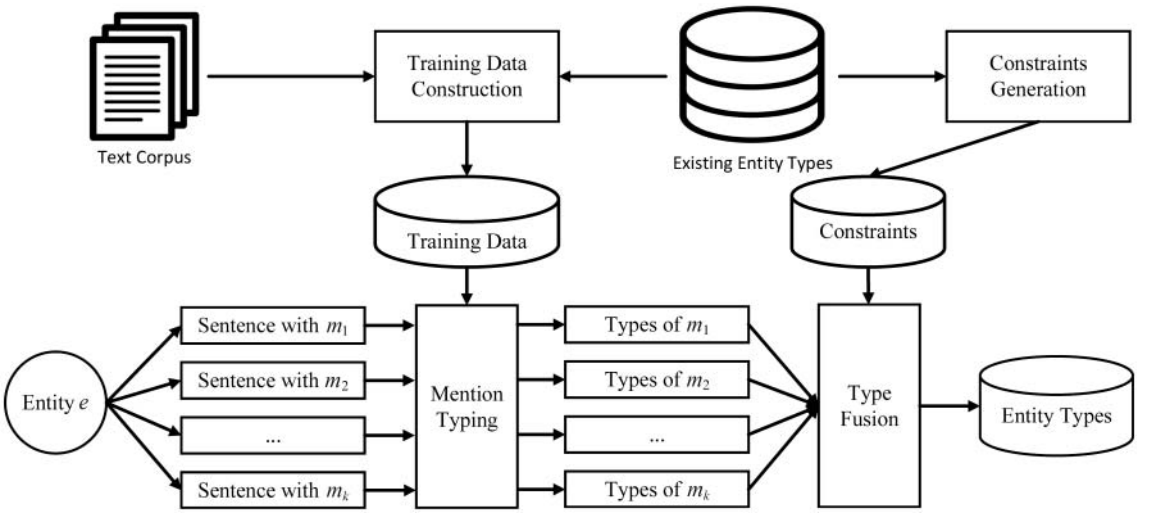


Figure 3. Framework of METIC [13], a multi-instance method for entity typing from unstructured features.

In mention typing step, we propose a neural network model, as shown in Figure 4. We first divide the sentence into three parts: the left context of the mention, the mention part and the right context of the mention. Each part of the sentence is fed into a parallel neural network with a similar structure: a word embedding layer and a BiLSTM (*bidirectional long short-term memory*) layer [17]. We then concatenate the outputs of these BiLSTM layers to generate the final output.

In type fusion step, we propose an integer linear programming model (ILP) to aggregate all the types derived from mentions of an entity to reduce the noise. ILP is an optimization model with constraints and all the variables required to be non-negative integers [18]. For each entity e, we first define a decision variable $x_{e,t}$ for every candidate type t. These variables are binary and indicate whether entity e belongs to type t or not. Our ILP model is as follows:

Maximize

$$\sum_{t\in T}\left(\max_{m\in M_e} P(t|m)-\theta\right)\times x_{e,t}$$

Subject to

$$\forall_{\mathrm{ME}(t_1,t_2)}\; x_{e,t_1}+x_{e,t_2}\le 1$$

$$\forall_{\mathrm{ISA}(t_1,t_2)}\; x_{e,t_1}-x_{e,t_2}\le 0$$

$$\forall_{t}\quad x_{e,t}\in\{0,1\}$$

where $\max_{m\in M_e} P(t|m)$ represents the maximum probability that one mention of the entity e belongs to type t, and θ is the threshold (In our experiment, we set the threshold as 0.5).

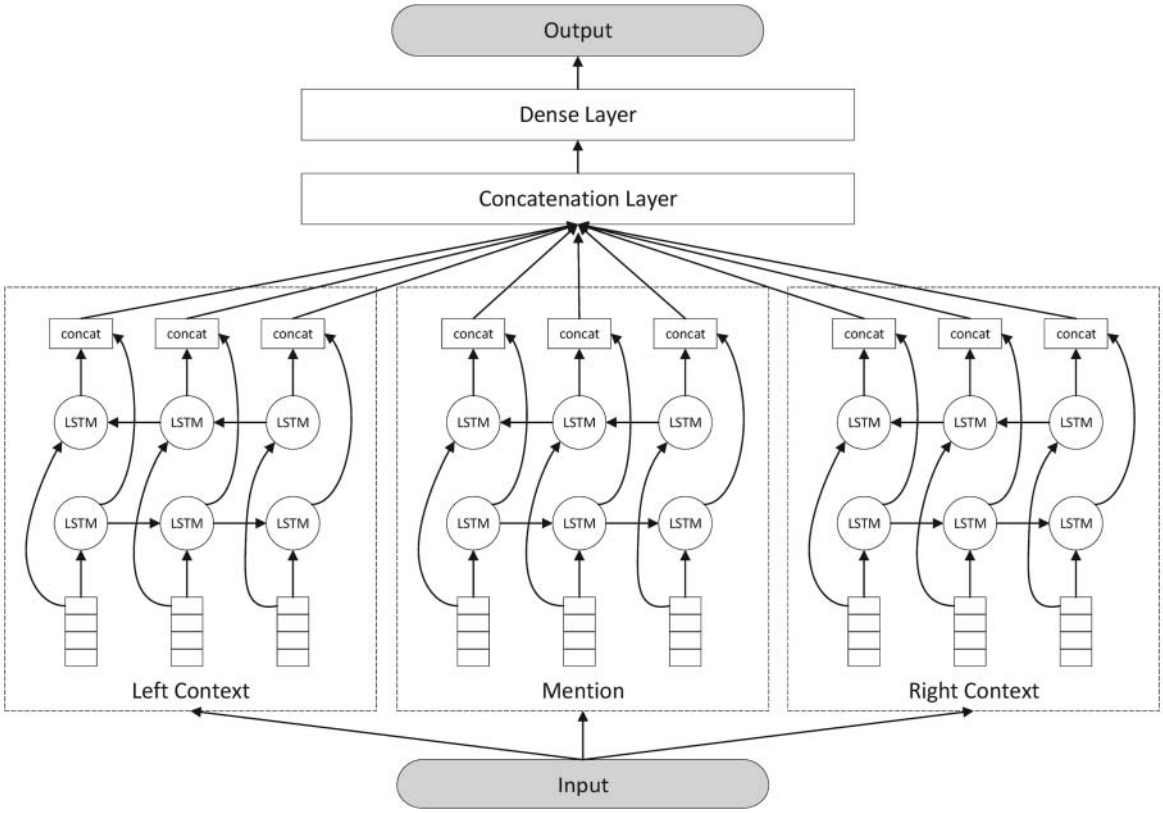


Figure 4. A neural network model for mention typing.

We propose two constraints for our ILP model: a *type disjointness* constraint and a *type hierarchy* constraint. The *Type disjointness* constraint claims that *an entity cannot belong to two semantically mutually exclusive types simultaneously*, such as *Person* and *Organization*. Types with no overlap in entities or an insignificant overlap below a specified threshold are considered to be disjoint [19]. The *Type hierarchy* constraint claims that *if an entity does not belong to a type t then it will certainly not belong to any of t's sub-types*. For example, an entity that does not belong to type *Artist* should not be classified to type *Actor*.

3.2 Entity Typing from Heterogeneous Features

In order to take advantage of the structured and unstructured features of the entity, we propose a new framework, *METIH* (**M**ulti-instance **E**ntity **T**yp**I**ng from **H**eterogeneous features), which is a modified version of the METIC. As shown in Figure 5, most of the components are the same as METIC, except the ones marked by the dotted line. Specially, in type fusion step, we treat the prediction result from structured features as a new instance of an entity, and use a new ILP model to aggregate those prediction results. The new ILP model is shown as follows:

Maximize

$$\sum_{t\in T}\left(\max\left(\max_{m\in M_e} P(t|m), \delta(t\in C_e)\right)-\theta\right)\times x_{e,t}$$

Subject to

$$\forall_{ME(t_1,t_2)}\ x_{e,t_1}+x_{e,t_2}\le 1$$

$$\forall_{ISA(t_1,t_2)}\ x_{e,t_1}-x_{e,t_2}\le 0$$

$$\forall_{t}\quad x_{e,t}\in\{0,1\}$$

where function $\delta(t\in C_e)$ is defined as follows:

$$\delta(t\in C_e)=\begin{cases}1, & \text{if type } t \text{ belongs to } C_e\\ 0, & \text{else}\end{cases} \tag{1}$$

$\max(\max_{m\in M_e} P(t|m),\ \delta(t\in C_e))$ represents the maximum probability that entity e belongs to type t, where $\max_{m\in M_e} P(t|m)$ is the maximum probability from unstructured features, while $\delta(t\in C_e)$ is the probability from structured features.

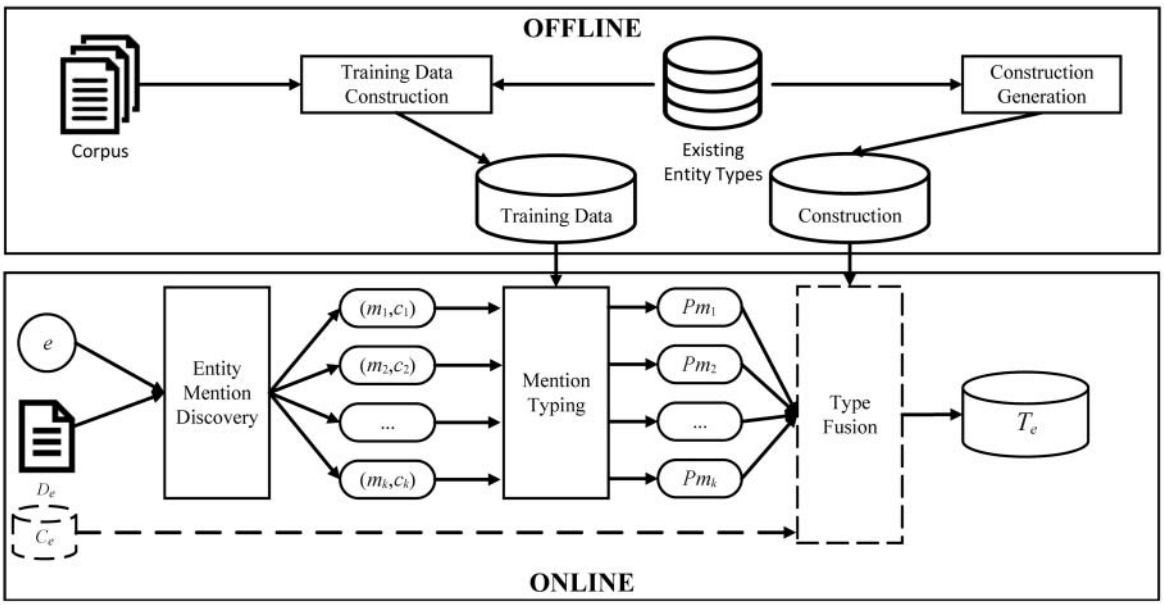


Figure 5. Framework of METIH, a multi-instance method for entity typing from heterogeneous features.

4. SLOT FILLING

Some entities' attribute values cannot be directly extracted due to missing information, so we extract the attribute values from text corpus. We cast it as a *slot filling* task, given an entity and an attribute, our goal is to extract the values from the description text of the entity. The state-of-the-art methods usually use supervised learning to build extractors for each predicate, and treat it as a *sequence tagging* problem.

However, the training samples for each predicate are unbalanced, and head predicates usually contain a large amount of training samples while long-tailed ones only contain few samples. In *CN-DBpedia*, we have extracted the values for head predicates. And in *CN-DBpedia2*, we focus on extracting the values for long-tailed predicates.

A naive solution for slot filling is to train a single predicate extractor for each predicate. The single predicate extractor is trained separately on data of different predicates. In the case of long-tailed predicates with insufficient training data, the single predicate extractors may not be fully trained. This impairs the performance. Therefore, we propose a **M**ultiple **P**redicate **E**xtractor with **P**rior **K**nowledge (**MPK**) model. The MPK model also takes the predicate as an input. We first pre-train a model by using all the head predicates, and then we fine-tune the model for each long-tailed predicates by transfer learning. Since most of the parameters in the model are shared along different predicates, it is feasible for the long-tailed predicate to utilize the abundant training data of head predicates by transfer learning.

Naturally, we expect to utilize the training samples from other predicates, which motivates us to develop a multiple predicate extractor structure. The multiple predicate extractor network structure is shown in Figure 6. The extractor is divided into five parts: 1) text embedding layer, 2) knowledge embedding layer, 3) text-knowledge attention layer, 4) encoder layer and 5) output layer.

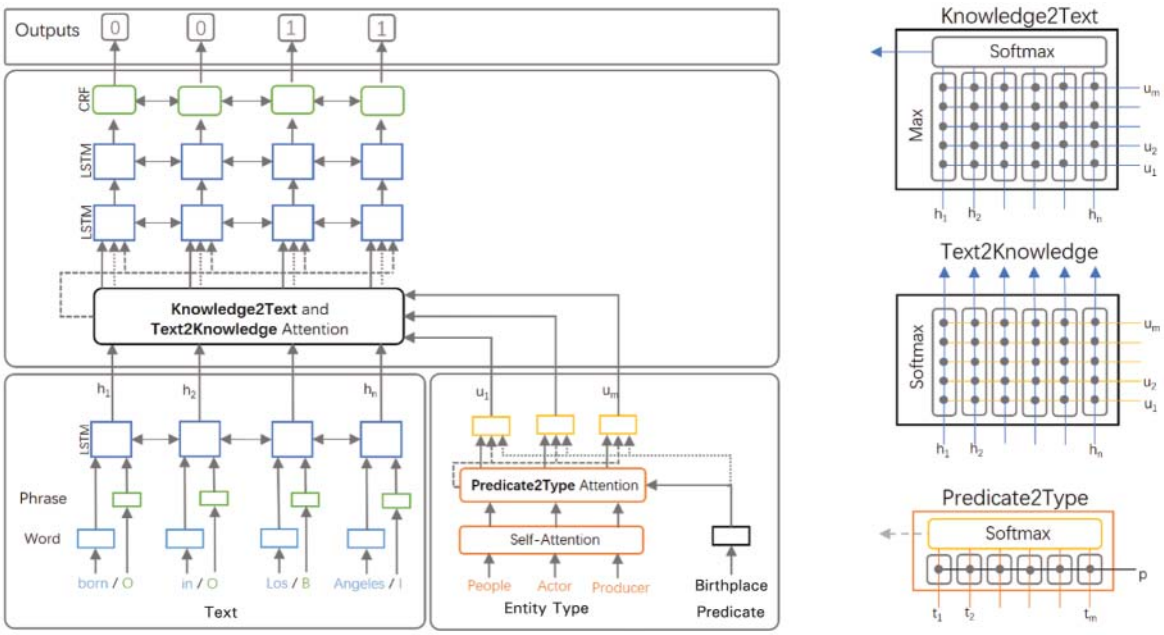


Figure 6. Network structures of the multiple-predicate extractor.

4.1 Text Embedding Layer

We first pre-process the sentences. The text embedding layer tends to capture the text features before extraction, including both the word and phrase information. Then we encode the concatenation of these embedded vectors with a BiLSTM layer.

4.2 Knowledge Embedding Layer

In this layer, we embed the prior knowledge as guidance for further information extracting. For example, they decide what kind of information should be extracted. The prior knowledge information includes the type from the entity (subject) and the predicate specifying the information to extract.

Type Encoder Layer. One entity may belong to multiple types, which implies different aspects of the entity. We use a self-attention layer to embed all the types of an entity, which is a multi-head attention mechanism proposed by [20]. Specifically, for each type, called the query, compute a weighted sum of all types, or keys, in the input based on the similarity between the query and key as measured by the dot product.

Predicate-Type Attention Layer. We combine the type and predicate information in this layer. The predicate and the type determine what to be extracted together. For example, with the entity type person and the predicate birthplace, we can determine that the task is to extract where the person was born, and which is similar to a query. However, an entity always has many noisy types (e.g., actor and producer in Figure 6). We use the predicate-to-type attention to select suitable types to form the query. Specifically, we use T and P to respectively denote the encoded types and predicate. The predicate-to-type similarity is computed as $S_0 \in \boldsymbol{R}^{1\times m}$. We then normalize the only row of S_0 by applying the softmax function, deriving a matrix $\overline{S}_0$. Then the predicate-to-type attention is computed as $A_0 = \overline{S}_0 \cdot T^T \in \boldsymbol{R}^{m\times d}$. The similarity function used here is the trilinear function [21]:

$$f(t,p) = W_0[t,p,t\odot p] \tag{2}$$

where $\odot$ is the element-wise multiplication and W_0 is a trainable variable.

The output of this module is an encoded knowledge sequence U, where each unit is $u = W_1[t,p,t\odot p, a_0 t]$, and t and p respectively denote an encoded type and the predicate embedding and a_0 is an attention value in A_0. Each unit u specifies an aspect of the extraction task, like the birthPlace of a person.

4.3 Text-Knowledge Attention Layer

This module uses the knowledge as a query to guide the extractor. We use H and U to denote the encoded text and knowledge. Then we adopt the attention similar to the BiDAF [21] to model the interaction between the text and the knowledge, including text-to-knowledge attention and knowledge-to-text attention.

The text-to-knowledge attention is constructed as follows. We first use Equation (2) to compute the similarities between each pair of encoded text and knowledge, rendering a similarity matrix $S \in \boldsymbol{R}^{m \times m}$. We then normalize each row of S by applying the softmax function, getting a matrix S. Then the text-to-knowledge attention is computed as $A = \overline{S} \cdot U^T \in \boldsymbol{R}^{m \times d}$.

We additionally use some form of knowledge-to-text attention. Specifically, we perform the maximum reduction on each column of S and compute the softmax function to get the text-to-knowledge attention $B = softmax\left(\max\left(S, 2\right)\right)$.

4.4 Model Encoder Layer

The input of this layer at each position is $[h,a,h \odot a,h \odot b]$, where a and b are respectively a row of attention matrix A and B. This module contains two BiLSTM layers.

4.5 Output Layer

In the output layer, we adopt a layer of Conditional Random Field, as it has been proven to be effective in the sequence tagging task [22].

4.6 Training

To tackle with insufficient training data for the long-tailed predicates, we hope that the model can benefit from abundant training samples of the head predicates. To achieve this, we let the model first learn how to extract the value of the head predicates, and then learn the extraction of the long-tailed predicates through transfer learning.

Specifically, we build extractors for each long-tailed predicate. The training processes include two steps. First, we pre-train the MPK model on training data of the top K head predicates. Then we fine tune the MPK model on the training data of the long-tailed predicate.

5. FACT VERIFICATION

Through the two steps of entity typing and slot filling, we can obtain a lot of facts from the text. But some of them are wrong, adding these errors to an existing knowledge base will reduce the quality of the knowledge base. Hence we need to verify the low confidence facts by human before adding them to the current knowledge base. The task of the verification process is as follows: given a piece of text and a fact extracted from the text, our goal is to let people judge whether this fact can be inferred from the text.

In general, it is very costly to verify all the facts by experts. To solve this problem, we propose a novel implicit crowdsourcing approach to ask users to verify those extracted facts. Our idea is inspired by CAPTCHA [23], which is a program that can generate and grade tests that: most humans can pass, but current computer programs cannot. Such a program can be used to differentiate humans from computers.

As shown in Figure 7, it asks users to type the distorted text seen from the image. If the typed characters match the characters in the image, it is considered verified.

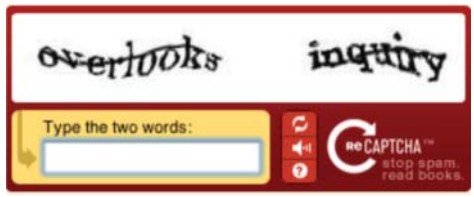


Figure 7. An example of CAPTCHA[②]. Note: In this case, the answer is "overlooks inquiry".

Specifically, we propose a different type of CAPTCHA, *rcCAPTCHA*, which is a reading comprehension test. Note that our goal is to let people verify whether a fact can be inferred from a piece of text. For each test, the reading passage is a piece of text, the question is generated from the fact, and the answer options are generated from both the text and the fact.

For different types of facts, we propose different ways to generate question and answer options. Given a fact derived from entity typing, such as (*e*, isA, *t*), where $e \in E$ is an entity and $t \in T$ is a type, we generate a question such as "*which type does* <**e**> *belong to?*", the answer options are *t* and all its siblings are in the type taxonomy, and the correct answer in this test is *t*. Since one entity may belong to many types, the entity *e* may belong to some sibling types of *t* according to other extracted facts. In this test, we exclude those types in the answer options. Given a fact derived from slot filling, such as (*e*, *a*, *v*), where $a \in A$ is an attribute and *v* is the attribute value, we generate a question such as "*what is the* <**a**> *of* <**e**>*?*", the answer options are all the words in the text (including the *v*), and the correct answer in this test is *v*. For each extracted fact, we generate some reading comprehension tests. When multiple users click on the correct answer and pass the test, the extracted facts are considered correct.

To verify the accuracy of the facts we extracted, we released a free rcCAPTCHA API[③] and deployed it on multiple websites. *CN-DBpedia* search engine[④] is one of those websites, and our rcCAPTCHA system is triggered when the number of user search exceeds a certain threshold. The search can only be continued if the user correctly answers the question from the rcCAPTCHA system. As shown in Figure 8, this is an instance of our system. The instance is used to verify whether the fact (*Galare Thong Tower, star rating, 3-star*) is correct. Based on this fact, the system generates a question (*What is the star rating of Galare Thong Tower?*) and a text containing the answer (*Galare Thong Tower is a* ***3-star*** *hotel located in Chiang Mai… It is a 9-minute drive from Wat Chiang Man …*). If majority of users click on "3-star" in the text, then the triple is considered correct.

② http://www.captcha.net/
③ http://kw.fudan.edu.cn/apis/supervcode/
④ http://kw.fudan.edu.cn/cndbpedia/

Figure 8. An example of our rcCAPTCHA system. Note: The word marked in yellow is the correct answer.

Our *rcCAPTCHA* system is triggered thousands of times per day on average, and each fact is verified by an average of 20 users. We randomly sampled 100 extracted facts that were verified to be correct with an accuracy of 100%. Although the current verification speed is slow, as our system is used by more and more people, our verification speed can be accelerated.

6. STATISTICS FOR CN-DBPEDIA2

We present the statistics of *CN-DBpedia*2 in this section. The number of entities and facts of *CN-DBpedia*2 is much larger than *CN-DBpedia*. By April 2019, *CN-DBpedia*2 contains about 16,024,656 entities and 228,499,155 facts, while *CN-DBpedia* only contains 10,341,196 entities and 88,454,264 facts. Table 2 shows the fact types in *CN-DBpedia*2, as well as changes compared to *CN-DBpedia*. The increase mainly comes from three aspects. Firstly, new sources of data have been added. Secondly, textual information is used. Thirdly, updating is implemented.

Table 2. Fact types in *CN-DBpedia*2, as well as changes compared to *CN-DBpedia*.

Rank	Rank count	Fact types	Fact quantity	Quantity change
1	0	Entity infobox	149,350,983	+108,210,921
2	+1	Entity types	45,065,595	+25,219,295
3	-1	Entity tags	25,923,300	+6,057,489
4	0	Entity information	8,016,829	+4,012,928
5	0	Entity sameAs	142,448	0

By using the new entity typing method, we found more types for entities in *CN-DBpedia*2. Table 3 shows the top 20 concepts and the number of entities they contain in *CN-DBpedia*2, as well as changes compared to *CN-DBpedia*. Because of the consideration of textual information, entities have gained more concepts, and the size of the concepts has increased significantly (except for the two concepts of companies and singles, the increase is mainly due to the extraction from the domain encyclopedia websites).

Table 3. The top 20 concepts and the number of entities included in *CN-DBpedia*2, as well as changes compared to *CN-DBpedia*.

Rank	Rank change	Type	Entity quantity	Quantity change
1	+1	Agent	8,771,817	+6,766,894
2	+7	Organization	7,019,610	+6,228,636
3	+9	Company	6,545,250	+6,128,240
4	-3	Work	4,786,508	+2,257,454
5	0	WrittenWork	2,918,856	+1,820,837
6	0	Book	2,742,963	+1,686,857
7	+9	Novel	1,504,497	+1,358,730
8	-5	Person	1,488,264	+270,276
9	-5	Place	1,401,132	+203,869
10	+7	MusicalWork	950,072	+699,196
11	+7	Single	897,631	+691,346
12	-4	PopulatedPlace	546,813	-69,209
13	-3	Settlement	404,672	-57,410
14	-5	ArchitecturalStructure	250,638	-241,942
15	-2	Species	247,723	+36,187
16	-2	Eukaryote	237,418	+29,647
17	+2	Software	230,900	+80,114
18	+2	Device	199,341	+75,976
19	+2	Athlete	167,293	+45,960
20	-5	Food	163,823	-14,866

Based on *CN-DBpedia*2, we also published some new APIs[5] for natural language understanding, including entity linking and question answering. By April 2019, these APIs had already been called 950 million times since they were published in December 2015.

7. CONCLUSION

In this paper, we released a new version of the knowledge base *CN-DBpedia*2, which is an extension of *CN-DBpedia*. Based on the existing knowledge base, we additionally exploit both structured and unstructured features to type entities together and propose a transfer learning strategy to extract the values for long-tailed predicates. Then we propose a novel implicit crowdsourcing approach to verify low confidence new facts and the facts verified as correct are added to the knowledge base. By April 2019, *CN-DBpedia*2 had contained about 16,024,656 entities and 228,499,155 facts, and the APIs had already been called 950 million times.

[5] http://kw.fudan.edu.cn/apis

AUTHOR CONTRIBUTIONS

This work was collaboration between all of the authors. Y. Xiao (shawyh@fudan.edu.cn) is the leader of the CN-DBpedia project. B. Xu (xubo@dhu.edu.cn) led the work and summarized the entity typing part. C. Xie (redreamality@gmail.com) and L. Chen (lhc825@gmail.com) summarized the slot filling part. J. Liang (l.j.q.light@gmail.com) summarized the fact verification part. B. Liang (liangbin@fudan.edu.cn) summarized the statistics part. All the authors have made meaningful and valuable contributions in revising and proofreading the resulting manuscript.

ACKNOWLEDGEMENTS

This paper was supported by National Key R&D Program of China (No. 2017YFC1201203), and sponsored by Shanghai Sailing Program (No.19YF1402300), and by the Initial Research Funds for Young Teachers of Donghua University (No. 112-07-0053019).

REFERENCES

[1] F.M. Suchanek, G. Kasneci, & G. Weikum. YAGO: A core of semantic knowledge. In: Proceedings of the 16th International Conference on World Wide Web, 2007, pp. 697–706. doi: 10.1145/1242572.1242667.

[2] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, & Z. Ives. DBpedia: A nucleus for a Web of open data. In: K. Aberer et al. (eds.) The Semantic Web. Berlin: Springer, 2007, pp. 722–735. doi: 10.1007/978-3-540-76298-0_52.

[3] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, & J. Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, 2008, pp. 1247–1250. doi: 10.1145/1376616.1376746.

[4] B. Xu, Y. Xu, J. Liang, C. Xie, B. Liang, W. Cui, & Y. Xiao. CN-DBpedia: A never-ending Chinese knowledge extraction system. In: International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, 2017, pp. 428–438. doi: 10.1007/978-3-319-60045-1_44.

[5] C. Xiong, R. Power, & J. Callan. Explicit semantic ranking for academic search via knowledge graph embedding. In: Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 1271–1279. doi: 10.1145/3038912.3052558.

[6] D. Yang, J. He, H. Qin, Y. Xiao, & W. Wang. A graph-based recommendation across heterogeneous domains. In: Proceedings of the 24th ACM International Conference on Information and Knowledge Management, 2015, pp. 463–472. doi: 10.1145/2806416.2806523.

[7] W. Cui, Y. Xiao, & W. Wang. KBQA: An online template based question answering system over Freebase. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI 2016), 2016, pp. 4240–4241. Available at: https://www.ijcai.org/Proceedings/16/Papers/640.pdf.

[8] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, … & C. Bizer. DBpedia—A large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web Journal 6(2) (2015), 167–195. doi: 10.3233/SW-140134.

[9] Q. Liu, K. Xu, L. Zhang, H. Wang, Y. Yu, & Y. Pan. Catriple: Extracting triples from Wikipedia categories. In: Asian Semantic Web Conference, 2008, pp. 330–344. doi: 10.1007/978-3-540-89704-0_23.

[10] B. Xu, Y. Zhang, J. Liang, Y. Xiao, S.-W. Hwang, & W. Wang. Cross-lingual type inference. In: International Conference on Database Systems for Advanced Applications, 2016, pp. 447–462. doi: 10.1007/978-3-319-32025-0_28.

[11] H. Adel, B. Roth, & H. Schütz. Comparing convolutional neural networks to traditional models for slot filling. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2016, pp. 828–838. Available at: https://www.aclweb.org/anthology/N16-1097.

[12] Y. Zhang, V. Zhong, D. Chen, G. Angeli, & C.D. Manning. Position-aware attention and supervised data improve slot filling. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017, pp. 35–45. doi: 10.18653/v1/D17-1004.

[13] B. Xu, Z. Luo, L. Huang, B. Liang, Y. Xiao, D. Yang, & W. Wang. METIC: Multi-instance entity typing from corpus. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 903–912. doi: 10.1145/3269206.3271804.

[14] M. Mintz, S. Bills, R. Snow, & D. Jurafsky. Distant supervision for relation extraction without labeled data. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2, 2009, pp. 1003–1011. doi: 10.3115/1690219.1690287.

[15] M. Surdeanu, J. Tibshirani, R. Nallapati, & C.D. Manning. Multi-instance multi-label learning for relation extraction. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, 2012, pp. 455–465. Available at: https://dl.acm.org/citation.cfm?id=2391003.

[16] L. Chen, J. Liang, C. Xie, & Y. Xiao. Short text entity linking with fine-grained topics. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 457–466. doi: 10.1145/3269206.3271809.

[17] S. Hochreiter, & J. Schmidhuber. Long short-term memory. Neural Computation 9(8) (1997), 1735–1780. doi: 10.1162/neco.1997.9.8.1735.

[18] J. Clarke, & M. Lapata. Global inference for sentence compression: An integer linear programming approach. Journal of Artificial Intelligence Research 31 (2008), 399–429. doi: 10.1613/jair.2433.

[19] N. Nakashole, T. Tylenda, & G. Weikum. Fine-grained semantic typing of emerging entities. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, 2013, pp.1488–1497. Available at: https://www.aclweb.org/anthology/P13-1146.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L.u. Kaiser, & I. Polosukhin. Attention is all you need. In: The 31st Conference on Neural Information Processing Systems (NIPS 2017), 2017, pp. 5998–6008. Available at: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

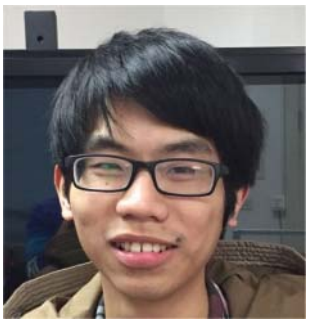
[21] M.J. Seo, A. Kembhavi, A. Farhadi, & H. Hajishirzi. Bidirectional attention flow for machine comprehension. arXiv preprint. arXiv:1611.01603, 2016.

[22] N. Reimers, & I. Gurevych. Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017, pp. 338–348. doi: 10.18653/v1/D17-1035.

[23] L. von Ahn, M. Blum, N.J. Hopper, & J. Langford. CAPTCHA: Using hard AI problems for security. In: International Conference on the Theory and Applications of Cryptographic Techniques, 2003, pp. 294–311. doi: 10.1007/3-540-39200-9_18.

AUTHOR BIOGRAPHY

Bo Xu is currently a Lecturer at School of Computer Science and Technology, Donghua University. He received his PhD degree from Fudan University in 2018. His research interests include knowledge base construction and applications. He has published serveral papers at major conferences including the International Joint Conference on Artificial Intelligence (IJCAI), the Conference on Information and Knowledge Management (CIKM), the European Conference on Principles of Data Mining and Knowledge Discovery (PKDD) and the International Conference on Database Systems for Advanced Applications (DASFAA). He won the Best Student Paper Award in the 31st National Database Conference (NDBC 2014).



Jiaqing Liang received his Bachelor's degree from School of Computer Science, Fudan University, China, in 2015. He is currently working towards the PhD degree at School of Computer Science, Fudan University, China. His research interests include knowledge bases and deep learning for text data.

Chenhao Xie is currently a PhD candidate at School of Computer Science, Fudan University. He is the co-founder of Shuyan Inc (http://shuyantech.com). His main research interests include information extraction and knowledge graph. He has published serveral papers at conferences including: the International Joint Conference on Artificial Intelligence (IJCAI), the Conference on Information and Knowledge Management (CIKM), and the International Conference on Data Mining (ICDM).

Bin Liang is a PhD candidate at School of Computer Science, Fudan University, China. He also completed his undergraduate studies at the Department of Computer Science and Technology, Fudan University. His research interests include data science, knowledge graph, recommender systems and social network mining.

Lihan Chen is a PhD candidate at School of Computer Science, Fudan University, China. His research interests include entity linking and information extraction.

Yanghua Xiao is a Professor at School of Computer Science, Fudan University. He is one of young "973" scientists. His research interests include big data management and mining, graph database and knowledge graph. Recently, he has published more than 70 papers in international leading journals and top conferences. He won the Best PhD Thesis Nomination of the Chinese Computer Federation (CCF) in 2010, CCF2014 Natural Science Award (second level), and ACM (CCF) Shanghai Distinguished Young Scientist Nomination Award.